Karlsruher Institut für Technologie

Institute of Information Security and Dependability (KASTEL)
Application-oriented Formal Verification
Prof. Dr. Bernhard Beckert

Bachelor thesis / Master thesis / Praxis der Forschung

A type system for your (least) favorite run-time error

Background. Pluggable type systems are a lightweight alternernative to more complex specification approaches, and a programmer-friendly way to find program errors at compile time and eliminate certain run-time exceptions.

```
@Interval(min=1, max=500) int i;
@Length(256) int[] arr;
public void foo(int) {
    int l = arr[i]; // Type error!
}
```

E.g., the Java Checker Framework has been used to create type systems for integer intervals (like in our example on the right), eliminating index-out-of-bound exceptions, for nullness, eliminating null-pointer exceptions, and even more complex concepts like SI units, eliminating programming bugs due to wrong unit conversions, or information flow, eliminating certain bugs to do with data leakage.

Task. Do you have ideas for common run-time errors—be they errors that occur in general programs or ones specific to a domain you specialize in—that could be avoided by a type system? If so, why not design, implement, and evaluate such a type system for your bachelor or master thesis, or during your *Praxis der Forschung*? The complexity and power of your type system will depend on the type of thesis.

Requirements. You have basic knowledge of programming-language design and type systems, e.g., from the lectures *Programming Paradigms* or *Formal Systems II*.

Contact
Florian Lanzinger lanzinger@kit.edu Office 50.34R227